通過テスト５ 解答

5 オーバーロードとコンストラクタ

1.

```
package e1;
class Article {
    String author;
    String subject;

    public Article() {}
    public Article(String author) { this("", author); }
    public Article(String author, String Subject){
        this.author  = author;
        this.subject = subject;
    }
}
public class Exec {
    public static void main(String[] args) {
        Article a = new Article();
    }
}
```

【解説】this(“”, author)および Exec の中で new Article()という呼び出しがあることから，引数を二つ持つコンストラクタと引数を持たないコンストラクタの 2 つをオーバーロードしなければならない.

2. D

【解説】しっかり理解するには準備のいる問題です。まず p.32 の「static と静的メンバ」を読んで、static の付いた静的メンバはオブジェクトの中にコピーされないことを理解します。次に、p.88 の「4.4 オブジェクト定義クラスの中の main メソッド」を読み返して、main メソッドを問題のように記述するのは便宜的な書き方であることも思いだしてください。

それから、本章の p.104「this とは」を見てください。this はオブジェクトが作成されたのち、オブジェクトのコンストラクタやメソッドの中で使います。このとき、this はそのオブジェクト自身の参照です。そのオブジェクト自身の参照を表す「代名詞」として this が使われます。

一方、main メソッドはオブジェクトの中にコピーされません。P.104 で説明しているように、オブジェクトの外に存在します。具体的には型情報としてクラスファイルがメモリーに読みこまれるとき（オブジェクトが作成される時など型の情報が必要)、その一部分と

して読み込まれます。オブジェクトの中にコピーされることはありません。ですからひとつだけ存在します。

`main` メソッドは `new` 演算子で作成しなくても存在しているのでオブジェクトではありません。したがって、`main` の中で `this` を使うとコンパイルエラーになります。それは、`this` はオブジェクトの中でしか使えないからです。

それから、一般のフィールド変数を「インスタンス変数」などと呼ぶのに対して、`static int size;` は静的メンバで「クラス変数」と呼ばれます（p.32 を参照）。型情報としてクラスファイルがメモリーに読みこまれるとき、その一部分として読み込まれます。オブジェクトの中にコピーされず、ひとつだけ存在します。

ただし、クラスのメンバ（構成要素）ですから、クラスのコンストラクタやメソッドから読み書きできます。いわば、コンストラクタとクラスメンバから共有される共有変数です。問題の `setSize` メソッドの中で `this.size = size;` としていますが、ここでの `this` は「このオブジェクトのメンバである」静的変数 `size` という意味になります。`this` を使ってこのオブジェクトとの関係だけを示すのでは誤りではありませんが、推奨されない書き方です。クラス変数はクラス全体の共有変数なので、クラス名を使って `Size.size` と書くのが正しい書き方です。「`Size` クラスの」静的変数 `size` という意味になります。

参考までに、問題を正しいプログラムに直すと次ぎのようになります。

```
public class Size {
        static int size;

        public void setSize(int size) {
            Size.size = size;
        }

        public static void main(String[] args) {
            Size s = new Size();
            s.setSize(50);
            System.out.println(Size.size); // 同じクラス内なので単に size としてもよい
        }
}
```

3. B, F

【解説】コンストラクタの引数の型と並び順に注意する．`0` は自動型変換されるので `int` でも `double` でも適合する．

4. ① B

【解説】引数に `double` を取るコンストラクタはすでに定義してあるので `A` は誤り．`B` は `int` なので正解，`C` は `String` を `double` にキャストできないので誤り．

② D

【解説】E と F の引数は double だか，double を引数にとるメソッドはすでに定義してあるので誤り．正解は D.引数に値がなくても D のように特定の値をフィールドに代入するような処理を書くことはできる．